Applying TOPSIS Method for Software Defined Networking (SDN) Controllers Comparison and Selection

Firas Fawzy Zobary[1,2](✉)

[1] Research Center of Network and Computing, Chongqing University of Posts and Telecommunications, Chongqing, People's Republic of China
firas_zobary@hotmail.com

[2] Faculty of Information Engineering, Damascus University, Damascus, Syrian Arab Republic

Abstract. Current traditional IP networks start to be complex as the demands of the users is ever-growing. Software Defined Network (SDN) is a new paradigm to ease the management of the network and make the network programmable by decoupling the control plane and forwarding plane (such as switch and router). A centralized controller is used to manage the control plane, and it interacts with forwarding plane using a standardized OpenFlow protocol. However, many controllers are used recently such as POX, Ryu, ONOS, and OpenDaylight. The important question is which is the best controller to use in our network and fits our network's goals? To answer this question, a decision making method is proposed in this paper. First, four SDN controllers are selected, and five criteria are analyzed to collect these controllers' properties. Then a Multi-Criteria Decision Making method named TOPSIS is used to rank the controllers and choose the best one. By applying this method, a comparative study is done to evaluate the four controllers in an environment of LAN topology, and "Ryu" controller is selected as the best one based on our criteria.

Keywords: Software defined networking · POX · Ryu · ODL · ONOS · MCDM

1 Introduction

Try to imagine the Internet as an old man who was living in the 50 s and moved to our life as it is now. That man will be shocked to find a lot of strange things like airplanes, mobile phones, and everything looks unfamiliar and very complex for his understanding. He will not be able to survive in our lifetime unless he starts to learn about these new things and adapt himself to use it. That's exactly how the internet is working now since its origin. When the internet was created it was very simple, not architected to use mobile or high-speed data transferring, and not secured well. It was designed for only exchanging information between end-nodes.

DOI: 10.1007/978-3-319-66628-0_13

Networks now are becoming a critical component of all the fields, and the goal now is to interconnect everything, through cloud computing, mobility or new concepts, like the internet of things.

However, in spite of its common adoption, traditional IP networking is still complex, very hard to manage. The switches, routers, and other devices implement a huge number of standardized protocols and proprietary interfaces that are still keep increasing.

Software Defined Network (SDN) is a new paradigm in which the network control is decoupled from the forwarding functions and both are communicating using a standardized OpenFlow protocol which enables the control plan to become directly programmable and the underlying infrastructure to be totally abstracted, so that makes this architecture ideal for today's applications which have a dynamic nature and demand high bandwidth.

The network will be programmable through software applications that run on the network operating system and interact with the devices from the data plane that becomes simple forwarding elements [1]. The architecture of SDN consists of two main planes, control plane and forward plane. The control plane is handled separately inside a controller which is one of the most important pieces of this technology's architecture. However, when we want to design our network using the concept of SDN, we must choose a suitable controller for our requirements. This decision problem is troublesome for many designers as it is difficult to define the right metrics, and the number of controllers keeps increasing. To solve this problem, we searched about the existing SDN controllers that are being used nowadays. We used websites, surveys, literature and any available resource providing this information. In the end, four controllers have been selected for our comparison to select the best one depending on properties we have already chosen. These controllers are: POX [2], Ryu [3], OpenDaylight [4], and ONOS [5].

As SDN controllers have different properties means that selecting a controller is a Multi-Criteria Decision Making (MCDM) problem [6]. Many MCDM methods are used for solving this kind of problems like Multiple Attribute Utility (MAUT), the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS), ELECTRE, AHP, etc. Recently researchers focus on AHP/ANP [13] based methods. Although these methods use experts' judgments and pairwise comparisons, but experts' intuitions and opinions conflict in uncertainty, and pairwise comparisons might be inconsistent with each other. SDN controllers comparison using AHP is tedious and time consuming. We used TOPSIS to select the best controller as it's more effective and quick for comparing and selecting the best alternative. In this paper, we proposed TOPSIS method to be combined with entropy to identify the criteria importance for SDN controller selection. Entropy is employed to get the weights of the criteria, and then TOPSIS method is used to rank the controllers and select the best controller that adjusts the decision maker's needs.

The outline of the paper is as follows: in Subsect. 2.1, we describe how the investigation about the controllers is done and which controllers' properties we'll focus on during the comparison. Subsect. 2.2 analyzes the candidate controllers based on these properties. In Sect. 3 we describe the comparison method that have been chosen and its steps. Section 4 describes the experimental work and the results of the method. Finally, a conclusion and future work is described in Sect. 5.

2 SDN Controllers Investigation

2.1 Methodology for the Investigation

To collect the properties of our selected SDN controllers, we have searched the white papers, surveys, and conferences that have described those controllers. Then we tried to search the official websites of each controller and other websites talking about them. We choose the following properties to use during the comparison between controllers.

- **Interfaces**: the controlling applications interact with the controller using northbound (NB) API and the controllers interact with the data plane using southbound (SB) API as shown in Fig. 1. NB uses many kinds of technologies such as REST [7]. SB APIs are divided into two categories: management and control. The management technologies are like OpenvSwitch Database Management Protocol (OVSDB) [8] and Simple Network Management Protocol (SNMP) [9]. The most famous SB control protocol is OpenFlow [10]. In forwarding plane, both physical and virtual switches can be used.

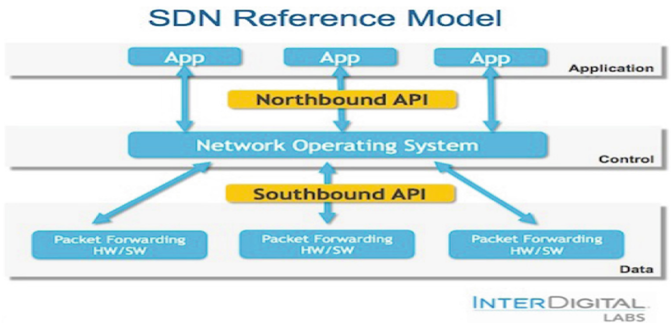

Fig. 1. SDN architecture

- The available documentation of each controller.
- **The OpenFlow version** that each controller uses. OpenFlow was developed during last years from version 1.0, and now we are in version 1.5, but still not all controllers support the same versions.
- **Programming language** by which each controller is used to develop applications. When we want to write our SDN application, we should use the programming language which is supported by the controller we choose for our network. In this paper, the priority for a programming language is its simplicity and how it's easy to learn when we want to develop our applications. Thus, we considered that the simpler programming language, the better the controller is.

- **Performance**: a comparison between controllers by their RTT delays time in switching mode was made. To do that, a tree topology was created using *mininet* tool [11]. Also, 16 hosts were created. Then ten ICMP packets were generated between the hosts 1&16, and the average RTT was calculated. These results were monitored during simple L2 learning switch phase where the controller links the source MAC address with the switch port from which the packet arrived and updates the flow table inside the switch by adding an entry to be used for forwarding any future packets holding that MAC address as a destination. The topology is illustrated in Fig. 2.

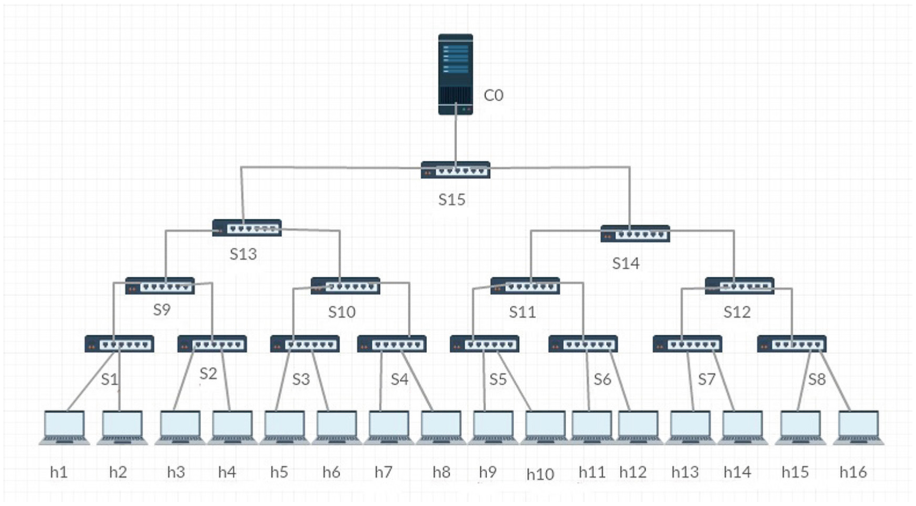

Fig. 2. SDN topology

2.2 SDN Controllers Overview

Four SDN controllers have been analyzed here: POX, Ryu, OpenDayLight, and ONOS.

A Python-based SDN controller, POX is a networking software platform. "POX is NOX's younger siblings" [2]. This controller is intended for faster development and is used to explore network virtualization, controller design, programming models, and to prototype new network applications. POX supports OpenFlow v1.0 as a southbound interface.

Ryu (the Japanese word for "flow") is a component-based SDN framework written in Python. It provides software components with well-defined API that make it easy for developers to create new network management and control applications. Ryu supports OpenFlow 1.0 to 1.4 and OVSDB as a southbound interface, and REST as a northbound interface.

OpenDaylight (ODL) is a modular Open SDN platform for networks of any size and scale. It's an open source controller, written in Java, and provides production-level performance and support. The goal of OpenDaylight project is to create robust code that covers most of the major components of the SDN architecture. The main drawback is the complexity and the fact that it takes time for learning to develop applications. It supports OpenFlow versions 1.0 & 1.3, and OVSDB in the southbound interface, and REST and OSGI as a northbound interface.

Another SDN controller is Open Network Operating System (ONOS). It is a software defined networking (SDN) OS for service providers that have scalability, high availability, and abstractions to make it easy to create apps and services. It's written in Java. Thus, it requires more time to learn, unlike POX or Ryu that were written in Python. ONOS supports OpenFlow version 1.0 and 1.3 and NetConf as a southbound interface.

3 The Decision Making Method

SDN controller selection is a Multi-criteria problem, where we will choose the best alternative among many alternatives considering a set of criteria and properties. Section 3.1 explains the concept of chosen MCDM method which is TOPSIS in our paper. Section 3.2 will explain how to use entropy to determine the importance for each criterion while Sect. 3.3 describes the steps of the method for SDN controller selection.

3.1 The Concept of TOPSIS Method

The main concept of TOPSIS method is to determine the positive ideal solution (PIS) and the negative ideal solution (NIS). Criteria in TOPSIS can be divided into two types: benefit and cost. Benefit means the large value is more valuable while cost criteria are vice versa. The PIS is the solution that maximizes the benefit criteria and minimizes the cost criteria while the NIS is doing the opposite by maximizing the cost criteria and minimizing the benefit criteria. After determining the PIS and the NIS, the technique will be to choose the alternative that has the shortest distance to the PIS and the farthest distance to the NIS. In general, any MCDM problem should have m alternatives and n criteria. The problem can be expressed in nxm matrix as follows:z

$$D = \begin{array}{c} \\ A_1 \\ A_2 \\ \vdots \\ A_m \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{C}_1, & \mathrm{C}_2, & \cdots & \mathrm{C}_n \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & \ddots & \cdots \\ x_{m1} & x_{m2} & \vdots & x_{mn} \end{bmatrix} \end{array}$$

$$W = (w_1, w_w, \cdots, w_n)$$

where $\mathrm{A}_1, \mathrm{A}_2, \ldots, \mathrm{A}_\mathrm{m}$ are alternatives, $\mathrm{C}_1, \mathrm{C}_2, \ldots, \mathrm{C}_\mathrm{n}$ are the criteria that we use, x_{ij} is the performance of alternative A_i under criterion C_j and w_j is the weight of criterion C_j, where $\sum_{\mathrm{j}=1}^{\mathrm{n}} \mathrm{w}_\mathrm{j} = 1$.

To perform the comparison across the criteria, matrix D should be transformed into dimensionless units by using the following equation:

$$P_{ij} = \frac{x_{ij}}{\sum_{i=1}^{m} x_{ij}}, j \in [1\ldots n] \tag{1}$$

3.2 Determination of Criteria Weights

When we want to compare SDN controllers and select the best one to use in our network, we'll face a problem that each criterion has a different importance. So it's necessary to determine the importance for each criterion which it's called "criterion's weight." There are many techniques to do that such as eigenvector method, entropy method, etc. Here, we will use entropy method to determine the weights for criteria.

In information theory, the entropy by Shannon [12] can be used to determine the disorder degree of the system is. Entropy weights method is based on the amount of information to determine the index's weight. Entropy value e_j can be calculated as:

$$e_j = \frac{-\sum_{i=1}^{m} P_{ij} ln(P_{ij})}{ln(m)}, i \in [1..m], j \in [1\ldots n] \tag{2}$$

Each criterion has different information, and the degree of that variation is calculated as:

$$d_j = 1 - e_j, j \in [1\ldots n] \tag{3}$$

Now we can calculate the weight for each criterion as:

$$w_j = \frac{d_j}{\sum_{j=1}^{n} d_j}, j \in [1\ldots n] \tag{4}$$

3.3 Selecting the Best Alternative

To do that we should first determine the PIS and the NIS. We labeled them as (A^+) and (A^-), respectively as:

$$A^+ = (P_1^+, P_2^+, \cdots, P_m^+) \tag{5}$$

$$A^- = \left(P_1^-, P_2^-, \cdots, P_m^-\right) \tag{6}$$

where:

$$P_j^+ = \left\{\max P_{ij}, j \in J^+; \min P_{ij}, j \in J^-\right\} \tag{7}$$

$$P_j^- = \left\{\min P_{ij}, j \in J^+; \max P_{ij}, j \in J^-\right\} \tag{8}$$

As J$^+$ and J$^-$ are the sets of benefit and cost criteria respectively. Next step is to calculate the distance between each alternative and the PIS (A$^+$) and NIS (A$^-$) as follows:

$$d_i^+ = \sqrt{\sum\nolimits_{j=1}^{n} w_j (P_j^+ - P_{ij})^2} \tag{9}$$

$$d_i^- = \sqrt{\sum\nolimits_{j=1}^{n} w_j (P_j^- - P_{ij})^2} \tag{10}$$

The last step is to calculate the relative degree of closeness of each alternative to the ideal solution. The relative degree of closeness for each alternative is defined as

$$\mu_i = \frac{d_i^-}{d_i^+ + d_i^-}, i \in [1, m] \tag{11}$$

The evaluation object is ranked according to the value of the relative degree of closeness. The best alternative is the one who has the highest μ.

4 The Best SDN Controller Selection (Experimental Results)

In this section, we'll apply previous MCDM method (TOPSIS) to select the best SDN controller among four controllers we choose for our comparison. Those controllers are: POX, Ryu, ODL, and ONOS, and the criteria for evaluating the alternatives are described as follows:

- Interfaces the controller uses to interact with applications and data plane (C1).
- Documentation for each controller (C2).
- OpenFlow version supported by a controller (C3).
- The programming language used by developers to develop applications (C4).
- Performance as the average RTT delays time for ICMP packets in milliseconds (C5).

The last criterion was studied in [14]. From the investigation that was done in Sect. 2.2, the results about each controller can be summarized in Table 1 shown below:

Table 1. Controllers' properties comparison

	Interfaces	Documentation	OpenFlow Version	Programming language	Average RTT (ms)
POX	OVSDB + OF	Poor	1.0	Python	20.76
Ryu	OVSDB + REST + OF	Medium	1.0 to 1.4	Python	11.86
ODL	REST + OSGI + OF + OVSDB	Good	1.3	Java	21.71
ONOS	REST + OSGI + OF + Netconf	Medium	1.0 & 1.3	Java	22.65

It's clear that the first four criteria are benefits as the higher value means the better controller, whereas the fifth one is cost criterion as the higher value of delay means the less performance from the controller.

The goal now is to convert the information in Table 1 into quantitative items that can be processed mathematically. To do that a range scale $[1 \rightarrow 4]$ will be used to refer to each qualitative item in the first four criteria, whereas the last criterion is already quantitative as we used the average RTT delay in milliseconds. We'll take into consideration that the high value in the scale $[1 \rightarrow 4]$ refers to a better performance according to what we have read in the available resources and what is the reputation about each candidate controller. After that, we'll have the decision matrix in Table 2.

Table 2. Decision matrix

	C_1	C_2	C_3	C_4	C_5
POX	1	1	1	3	20.76
RYU	2	2	4	3	11.86
ODL	3	3	1	4	21.71
ONOS	3	2	2	4	22.65

In Table 3. The normalized dimensionless matrix, PIS (A^+), NIS (A^-), entropy and the weights are shown. The last step is shown in Table 4, where we calculated the distances d_i^+ and d_i^-, then the relative degree of closeness μ_i. Based on μ_i values the alternatives are ranked and according to our method and criteria the best SDN controller is Ryu.

Table 3. Normalized matrix, entropy (e), weights (w), PIS, and NIS

	C_1	C_2	C_3	C_4	C_5
POX	0.111	0.125	0.125	0.214	0.27
RYU	0.222	0.25	0.5	0.214	0.154
ODL	0.333	0.375	0.125	0.286	0.282
ONOS	0.333	0.25	0.25	0.286	0.294
e	0.945	0.953	0.875	0.992	0.98
w	0.216	0.184	0.49	0.031	0.078
A^+	0.333	0.375	0.5	0.286	0.154
A^-	0.111	0.125	0.125	0.214	0.294

In the end, Fig. (3) shows the chart of alternatives and their distances to the PIS (d+) and NIS (d−) and the relative degree of closeness (μ).

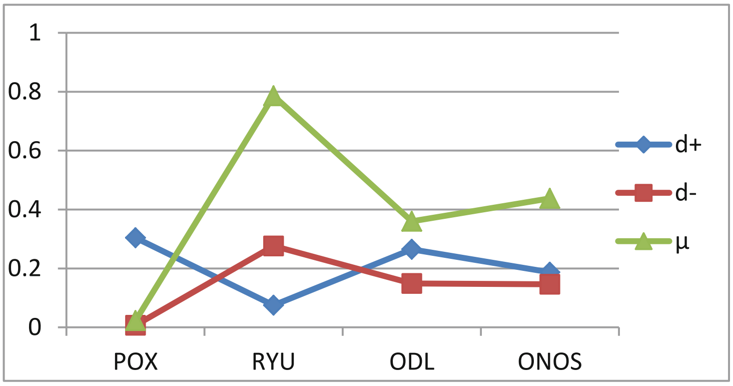

Fig. 3. Final results

Table 4. Distances and relative closeness

	d^+	d^-	μ
POX	0.304	0.007	0.023
Ryu	0.075	0.276	0.786
ODL	0.265	0.149	0.36
ONOS	0.187	0.146	0.438

5 Conclusion and Future Work

In this paper, we proposed TOPSIS method with entropy weights to compare and select the best controller among four chosen SDN controllers based on five criteria in an environment of LAN topology. The result showed that the proposed method is simple and flexible. In the end, our alternatives were ranked as Ryu, ONOS, ODL, and POX which means that Ryu controller is the best controller according to the criteria we studied as it had the shortest distance to the PIS and the farthest distance to the NIS. In the future works, more criteria can be added such as hardware system requirements of each controller. Also, a larger network scale such as datacenter with much more network devices can be added to the topology and the same comparative study would be applied to check if the results will stay the same or a different controller will be chosen, and also we can apply another MCDM methods to see if Ryu controller will stay the best alternative or using pairwise comparisons will affect the selection results.

References

1. Kreutz, D., Ramos, F.M.V., Esteves Verissimo, P., Esteve Rothenberg, C., Azodolmolky, S., Uhlig, S.: Software-defined networking: a comprehensive survey. Proc. IEEE **103**(1), 14–76 (2015)

2. About POX | NOXRepo. http://www.noxrepo.org/pox/about-pox/. Accessed 26 Apr 2016
3. Ryu SDN Framework. http://osrg.github.io/ryu/. Accessed 26 Apr 2016
4. The OpenDaylight Platform | OpenDaylight. https://www.opendaylight.org/. Accessed 26 Apr 2016
5. ONOS - A new carrier-grade SDN network operating system designed for high availability, performance, scale-out. http://onosproject.org/. Accessed 26 Apr 2016
6. Ehrgott, M., Gandibleux, X.: Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys. Kluwer Academic Publishers, Boston (2002)
7. Fielding, R.T.: Architectural styles and the design of network-based software architectures, University of California, Irvine (2000)
8. Pfaff, B., Davie, B.: The Open vSwitch Database Management Protocol (2013)
9. hjp: doc: RFC 1157: Simple Network Management Protocol (SNMP). http://www.hjp.at/doc/rfc/rfc1157.html. Accessed 26 Apr 2016
10. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
11. Mininet Overview - Mininet. http://mininet.org/overview/. Accessed 26 Apr 2016
12. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mob. Comput. Commun. Rev. **5**(1), 3–55 (2001)
13. Wei, C.C., Chien, C.F., Wang, M.J.J.: An AHP-based approach to ERP system selection. Int. J. Prod. Econ. **96**(1), 47–62 (2005)
14. Huang, J.: Combining entropy weight and TOPSIS method for information system selection. In: 2008 IEEE Conference on Cybernetics and Intelligent Systems, pp. 1281–1284. IEEE, September 2008